American Computer Science League

2022-2023 ● Contest 3: Create a Tree ● Senior Division

PROBLEM: Given a string to create 2 parallel arrays, process the letters in the string from left to right, inserting each letter into the first array in alphabetical order, one letter at a time. During this process, each letter is assigned an integer value in the second array as follows:

1. The first letter has a value of 0.
2. If the new letter is the first or last in the array, its value is one more than the value of the adjacent letter.
3. Otherwise, it is one more than the larger value of the two adjacent values. If the letter is already in the array, the new letter is placed before the existing letter.

Once the two arrays are created, output two different strings separated by a single space.
To create the first string, add letters to the string as they are found starting with the letter that has a value of 0.

1. Look to its left to find a letter with the next consecutive value. If it exists, add it to the string.
2. Stop the recursion if no more letters are available or a letter with a lower value is found.
3. Look to the current letter's right to find a letter with the next consecutive value. If it exists, add it to the string.

Do steps 1-3 recursively for each letter found to the left of the letter with a value of 0.
Find the 1 to the right of the letter with a value of 0.
Do steps 1-3 recursively for each letter found to the right of the letter with a value of 0.

To create the second string, access the letters in the same consecutive order as above looking first to the left and then to the right. Stop the recursion if you find a letter such that nothing that can be found after that. Nothing can be found if either there are no more letters or a letter with a smaller value is found first. Add that letter to the string. Return to the previous letter to continue the recursion.

EXAMPLE #1: Here is how the array is built for the string **PYTHONN**:
The letter P illustrates rule 1; Y and H illustrate rule 2; the T, O and Ns illustrate rule 3; and the final N illustrates what happens if there is a duplicate letter.

P	*Letters*	P								
	Values	0								

Y	*Letters*	P	Y							
	Values	0	1							

T	*Letters*	P	T	Y						

	Values	0	2	1						
H	*Letters*	H	P	T	Y					
	Values	1	0	2	1					
O	*Letters*	H	O	P	T	Y				
	Values	1	2	0	2	1				
N	*Letters*	H	N	O	P	T	Y			
	Values	1	3	2	0	2	1			
N	*Letters*	H	N	N	O	P	T	Y		
	Values	1	4	3	2	0	2	1		

The steps used to find the first and second string are as follows:

- Access the P with a value of 0, then the H with a value of 1 to the left of the P. There are no letters that can be found to the left of H. Find the O to the right of H with a value of 2, then N with a value of 3 to the left of O, then the N with a value of 4 to the left of the N with a value of 3. There are no letters that can be found with a value greater than 4 to the left or the right of the N with a value of 4 so stop the recursion. There are also no values greater than 3 to the right of the N with a value of 3 and no values greater than 2 to the right of the O. To the right of the P, find the Y with a value of 1, and finally the T to its left with a value of 2.

- In the first string, add each letter as it is found. The result is PHONNYT.

- For the second string, after the P, find the H, then the O, then the N with a value 3, then the N with a value of 4. Since there is nothing else to its left or right that can be found, add the N with a value of 4 to the string. Return to the previous letter which is the N with a value of 3 and add it to the string. Return to the O, add it to the string, return to the H, and add it to the string. There are no more letters that can be found to the left of P so look to the right of P. Find the Y, then the T. Therefore, add the T to the string since there are no more letters that can be found to both its left and right, then the Y, and finally the P. The result is NNOHTYP.

The output for these arrays is: PHONNYT NNOHTYP.

EXAMPLE #2: The string **BINARYSEARCHTREE** results in the following array:

Letters	A	A	B	C	E	E	E	H	I	N	R	R	R	S	T	Y
Values	2	1	0	3	5	4	2	3	1	2	5	4	3	5	6	4

Access the letters for both strings as follows: On the left of P with a value of 0, find the A with a value of 1, then the A with a value of 2 and stop. On the right of P with a value of 0, find the I with a value of 1, then the E to the left with a value of 2, then the C to the left with a value of 3, then the E to the right with a value of 4, then the E to the left with a value of 5, then the H with a value of 3 to the right of the E with a value of 2 since you have done everything to the left of the E with a value of 2 and now must look to its right. Continue on the right side of the I with a value of 1. The output for these arrays is: BAAIECEEHNRRRYST AAEECHERRTSYRNIB.

INPUT: Input a string to use in building the two arrays. The string will be no more than 80 characters containing all capital letters.

OUTPUT: Output a string representing the first and second strings separated by a single space.

SAMPLE INPUT:

1. PYTHONN
2. BINARYSEARCHTREE
3. CORONAVIRUS
4. FINALSFORACSL
5. HACKERRANKPLATFORM

SAMPLE OUTPUT:

1. PHONNYT NNOHTYP
2. BAAIECEEHNRRRYST AAEECHERRTSYRNIB
3. CAOONIRRVUS AINORSUVROC
4. FAAFCINLLSORS ACFALLSROSNIF
5. HAAACEFKKRRNLMPORT AAFECAKMLORPNRTRKH

2022-2023 ● Contest 3: Create a Tree ● Senior Division

TEST DATA

TEST INPUT:

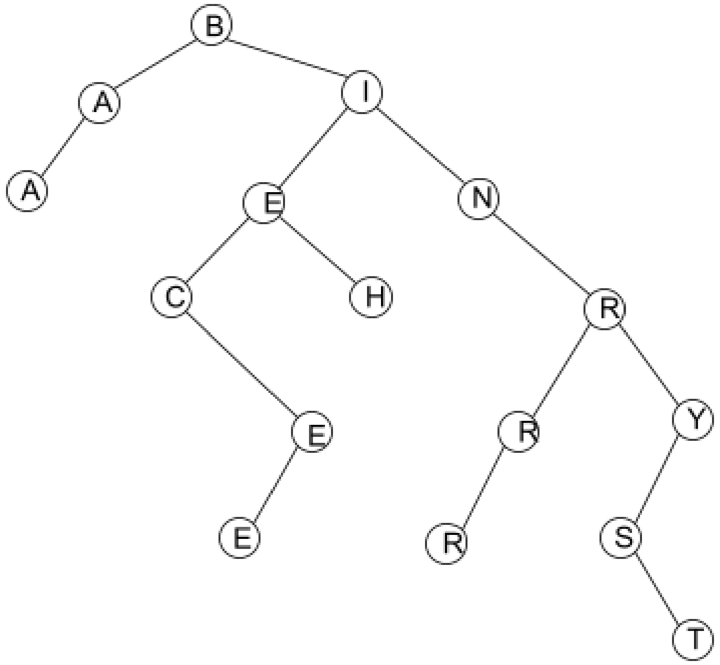
1. MOTHER
2. ACSLCONTEST
3. SUPERCALIFRAGILISTIC
4. JAVAPROGRAMMING
5. ABDFHKMOQTVWYZ

TEST OUTPUT:

1. MHEOTR EHRTOM
2. ACCSLEONSTT CENSOLTTSCA
3. SPECAACLIFGIIILRRSUT ACACIIIGFLILERSRPTUS
4. JAAAGGIVPOMMNRR AAGIGAMNMORRPVJ
5. ABDFHKMOQTVWYZ ZYWVTQOMKHFDBA

FOR INFORMATION PURPOSES ONLY

The procedure above is a way to create a binary search tree. The values in the array represent the depth of the nodes. The tree below is the result:



The letters printed are the preorder and postorder traversals of the tree. The preorder traversal is root→left→right recursively which produces BAAIECEEHNRRRYST. The postorder traversal is left→right→root recursively which produces AAEECHERRTSYRNIB.